IN THE CLAIMS

For the convenience of the Examination, the amending claims are as follows:

1-6. (Canceled)

7. (Previously Presented) A method of software pipelining for improving efficiency of loop handling, the method comprising:

checking for availability of rotating registers to hold computed values that are live across multiple stages in a software-pipelined loop; and

spilling and filling the computed values held in rotating registers in a software-pipelined loop using rotating stack memory locations for rotating registers, when there are no rotating registers available to hold the computed values, wherein the number of the rotating stack memory locations used for spilling and filling the computed values is equal to the number of simultaneous live values generated by the rotating register.

8. (Original) The method of claim 7, wherein the computed values are Floating Point (FP) values.

9. (Original) The method of claim 7, wherein the rotating registers are FP rotating registers.

10. (Previously Presented) A method of software pipelining for improving efficiency of loop handling, the method comprising:

checking for availability of FP rotating registers to hold FP computed values that are live across multiple stages in a software-pipelined loop; and

spilling and filling the computed values using rotating integer registers for holding addresses of stack memory locations when there are no FP rotating registers available to hold the computed value, the spilling and filling the computed values comprising:

checking for availability of N+1 rotating integer registers, wherein N is number of stages a computed value that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed value in stack memory locations whose addresses are held in corresponding N+1 rotating integer registers, when the N+1 rotating integer registers are available.

11. (Canceled)

12. (Previously Presented) The method of claim 10, wherein spilling and filling the computed value comprises:

storing the computed value in the stack memory locations whose addresses are held in corresponding N+1 rotating integer registers; and

loading from the stack memory locations whose addresses are held in corresponding N+1 rotating integer registers based on number of stages between when the loading occurs from the storing of the corresponding computed value.

13. (Original) The method of claim 10, wherein target registers for filling could be any available FP registers.

14. (Previously Presented) A method of software pipelining for improving efficiency of loop handling, the method comprising:

using post-incremented memory operations for spilling and filling of live computed values, held in a FP rotating register, that are live across multiple stages in a software-pipelined loop, using non-rotating registers, when there are no rotating integer registers available to hold rotating stack memory locations,

checking for availability of N+1 non-rotating integer registers available for spilling and filling, wherein N is a number of stages a computed value that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed values in stack memory locations whose addresses are held in corresponding N+1 non-rotating integer registers, when the N+1 non-rotating registers are available.

15. (Original) The method of claim 14, wherein using the non-rotating registers comprises:

using the non-rotating integer registers.

16-17. (Canceled)

18. (Previously Presented) A method of software pipelining for improving efficiency of loop handling, the method comprising spilling and filling of live computed values, held in a rotating register, that are live across multiple stages in a software-pipelined loop, using two non-rotating integer registers, when there are no FP rotating registers available and when there are no rotating integer registers available for holding rotating stack memory locations, and when there are not enough non-rotating integer registers available for holding rotating stack memory locations.

19. (Original) The method of claim 18, wherein the two non-rotating registers do not have to be contiguous.

20. (Original) The method of claim 18, wherein the rotating stack memory locations have to be contiguous and in descending order.

21. (Previously Presented) A method of software pipelining for improving efficiency of loop handling, the method comprising:

checking for availability of rotating integer registers and non-rotating integer registers, to spill and fill computed values held in a FP rotating register, that are live across multiple stages in a software-pipelined loop;

spilling and filling the computed values, held in a FP rotating register, using the rotating integer registers to hold rotating stack memory locations, when there are no FP rotating registers available to hold the computed values;

spilling and filling the computed values, held in the FP rotating register, using non-rotating registers to hold the rotating stack memory locations, when there are no FP rotating registers to hold the computed values and further when there are no rotating integer registers available for holding rotating stack memory locations; and

spilling and filling the computed values held in the FP rotating register, using two non-rotating integer registers to hold the rotating stack memory locations, when there are no FP rotating registers to hold the computed values, where there are no rotating integer registers available, and further when there are only a few non-rotating integer registers available for holding rotating stack memory locations.

22. (Original) The method of claim 21, wherein spilling and filling the computed values using the rotating integer registers comprises:

checking for availability of N+1 rotating integer registers, wherein N is number of stages a computed value that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed values in stack memory locations whose addresses are held in corresponding N+1 rotating integer registers, when the N+1 rotating integer registers are available.

23. (Original) The method of claim 21, wherein spilling and filling the computed values using non-rotating integer registers comprises:

checking for availability of N+1 non-rotating integer registers available for spilling and filling, wherein N is a number of stages a computed value that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed values in stack memory locations whose addresses are held in corresponding N+1 non-rotating integer registers, when the N+1 non-rotating registers are available.

24. (Original) The method of claim 21, wherein the two non-rotating registers do not have to be contiguous.

25. (Original) The method of claim 21, wherein the rotating stack memory locations have to be contiguous and in descending order.

26. (Canceled)

27. (Previously Presented) The article comprising a computer-readable medium which stores computer-executable instructions of claim 26, wherein spilling and filling computed values that are live across multiple stages in a software-pipelined loop using rotating stack memory locations comprises:

checking for availability of rotating integer registers and non-rotating integer registers, to spill and fill computed values held in a FP rotating register, that are live across multiple stages in a software-pipelined loop;

spilling and filling the computed values, held in a FP rotating register, using the rotating integer registers to hold the rotating stack memory locations, when there are no FP rotating registers available to hold the computed values;

spilling and filling the computed values, held in the FP rotating register, using the non-rotating registers to hold the rotating stack memory locations, when there are no FP rotating registers to hold the computed values and further when there are no rotating integer registers available for holding rotating stack memory locations; and

spilling and filling the computed values held in the FP rotating register, using two non-rotating integer registers to hold the rotating stack memory locations, when there are no FP rotating registers to hold the computed values, where there are no rotating integer registers available, and further when there are only a few non-rotating integer registers available for holding rotating stack memory locations.

28. (Original) The article comprising a computer-readable medium which stores computer-executable instructions of claim 27, wherein spilling and filling the computed values using the rotating integer registers comprises:

checking for availability of N+1 rotating integer registers, wherein N is number of stages a computed values that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed values in stack memory locations whose addresses are held in corresponding N+1 rotating integer registers, when the N+1 rotating integer registers are available.

29. (Original) The article comprising a computer-readable medium which stores computer-executable instructions of claim 27, wherein spilling and filling the computed values using non-rotating integer registers comprises:

checking for availability of N+1 non-rotating integer registers available for spilling and filling, wherein N is a number of stages a computed value that needs to be spilled is live in the software-pipelined loop; and

spilling and filling the computed values in stack memory locations whose addresses are held in corresponding N+1 non-rotating integer registers, when the N+1 non-rotating registers are available.

30. (Previously Presented) A system comprising:

a bus;

a processor coupled to the bus;

a memory coupled to the processor; and

a network interface coupled to the processor and the memory, wherein the processor to spill and fill multiple computed values, in a register, that are live across multiple stages in a software-pipelined loop, by performing:

checking for availability of rotating integer registers and non-rotating integer registers, to spill and fill computed values held in a FP rotating register, that are live across multiple stages in a software-pipelined loop;

spilling and filling the computed values, held in a FP rotating register, using the rotating integer registers to hold rotating stack memory locations, when there are no FP rotating registers available to hold the computed values;

spilling and filling the computed values, held in the FP rotating register, using the non-rotating registers to hold the rotating stack memory locations, when there are no FP rotating

registers to hold the computed values and further when there are no rotating integer registers available for holding rotating stack memory locations; and

spilling and filling the computed values held in the FP rotating register, using two non-rotating integer registers to hold the rotating stack memory locations, when there are no FP rotating registers to hold the computed values, where there are no rotating integer registers available, and further when there are only a few non-rotating integer registers available for holding rotating stack memory locations.

31. (Original) The system of claim 30, wherein the processor checks for availability of N+1 rotating integer registers, wherein N is number of stages a computed value that needs to be spilled is live in the software-pipelined loop, and spills and fills the computed values in stack memory locations whose addresses are held in corresponding N+1 rotating integer registers, when the N+1 rotating integer registers are available.

32. (Original) The system of claim 30, wherein the processor checks for availability of N+1 non-rotating integer registers available for spilling and filling, wherein N is a number of stages a computed value that needs to be spilled is live in the software-pipelined loop, and spills and fills the computed values in stack memory locations whose addresses are held in corresponding N+1 non-rotating integer registers, when the N+1 non-rotating registers are available.